Deep Neural Network for Youtube Recommendations

Paul Covington, Jay Adams, Emre Sargin
Google

RecSys, 2016

Outline

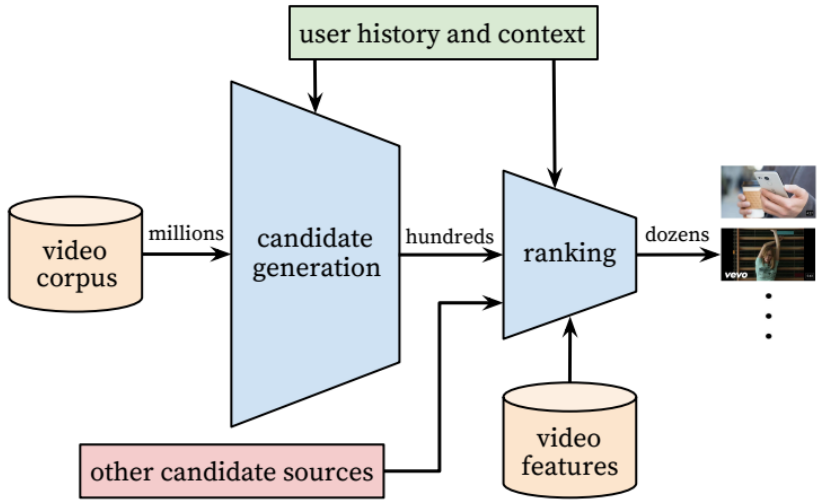
1. Introduction
2. System Overview
3. **Candidate Generation Network**
    - 3.1. Recommendation as Classification
    - 3.2. Model Architecture
    - 3.3. Heterogeneous Signals
    - 3.4. Label and Context Selection
    - 3.5. Experiments with Feature and Depth
4. **Raking Network**
    - 4.1. Feature Representation
    - 4.2. Modeling Expected Watch Time
    - 4.3. Experiments with Hidden Layers
5. Conclusion

1. Introduction

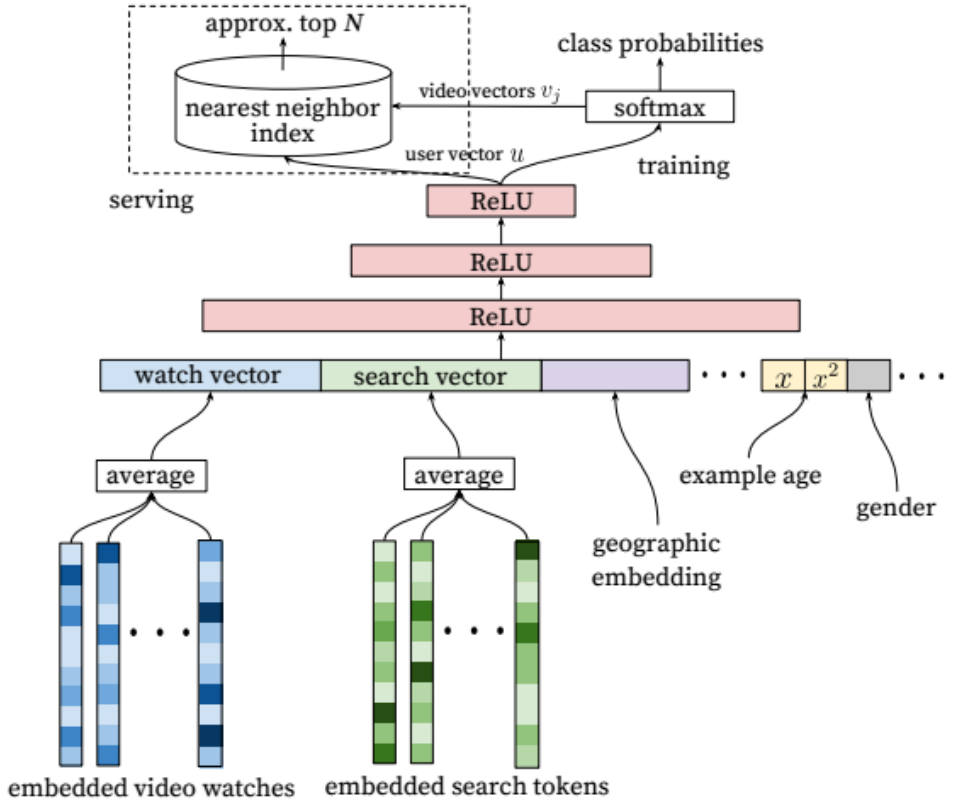
- Goal: help more than a billion users discover personalized content from an ever-growing corpus of videos.
- Recommending YouTube videos is extremely challenging from ***Scale***, ***Freshness***, ***Noise***.
- Deep learning: learn approximately **one billion parameters** and are trained on **hundreds of billions of examples**.

2. System Overview

- The system is comprised of two neural networks.
- ***Candidate generation*** *network:*
  - Provide broad personalization via collaborative filtering.
  - The similarity between users is expressed in terms of coarse features such as IDs of video watches, search query tokens and demographics.
- ***Ranking*** *network:*
  - Assign a score to each video according to a desired objective function using a rich set of features describing the video and user. The highest scoring videos are presented to the user, ranked by their score.



3. Candidate Generation



During candidate generation, the enormous YouTube corpus is winnowed down to hundreds of videos that may be relevant to the user.

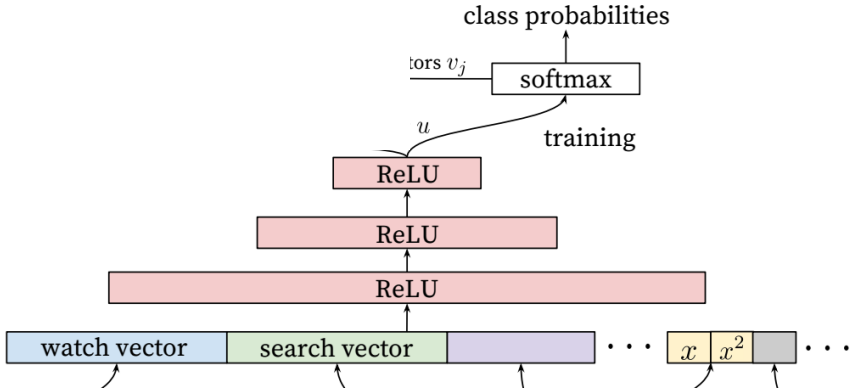
3.1 Recommendation as Classification

- Pose recommendation as extreme multiclass classification where the prediction problem becomes accurately classifying a specific video watch w_t at time t among millions of videos i (classes) from a corpus V based on a user U and context C.

$$P(w_t = i|U, C) = \frac{e^{v_i u}}{\sum_{j \in V} e^{v_j u}}$$

- The task of the deep neural network is to learn user embeddings u as a function of the user's history and context that are useful for discriminating among videos with a softmax classifier.

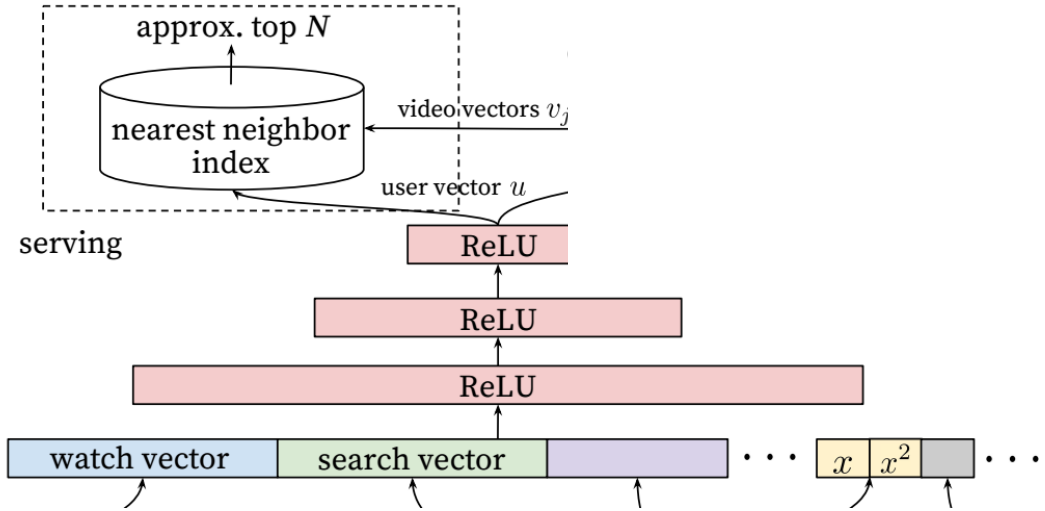
- To efficiently train such a model with millions of classes, sample negative classes from the background distribution ("candidate sampling").



Training

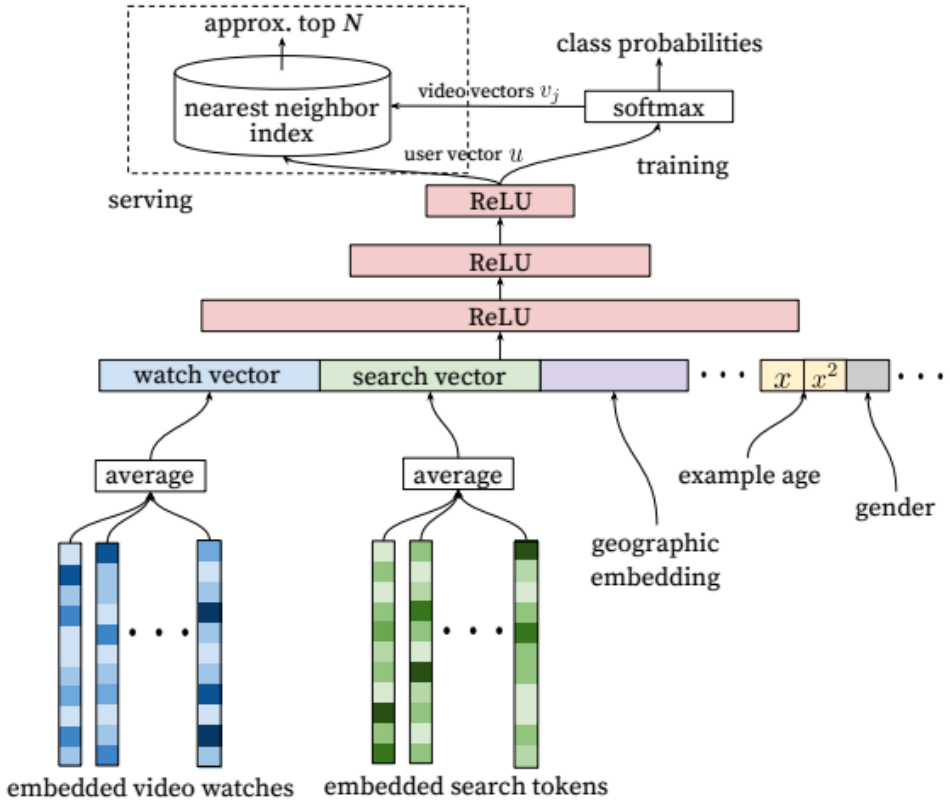
3.1 Recommendation as Classification

- At serving time we need to compute the most likely N classes (videos) in order to choose the top N to present to the user.
- Score millions of items under a strict serving latency of tens of milliseconds.
- Collaborative filtering: reduce to a *kNN* search



Serving

3.2 Model Architecture



3.3 Heterogeneous Signals

- Search history
- watch history
- Demographic features
- **“Example Age” Feature**

- Users prefer fresh content.
- Recommending recently uploaded (“fresh”) content is extremely important.

- Feed the **age** of the training example as a feature during training.

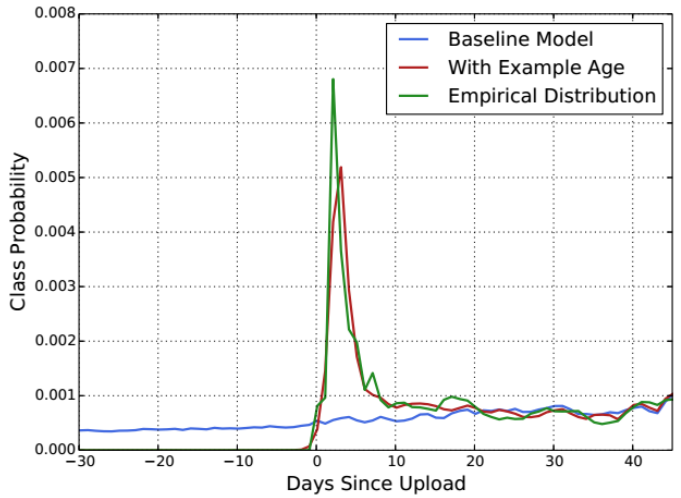
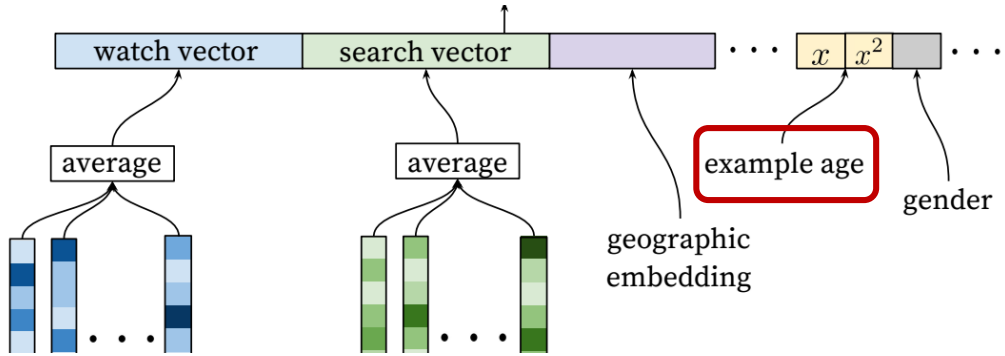


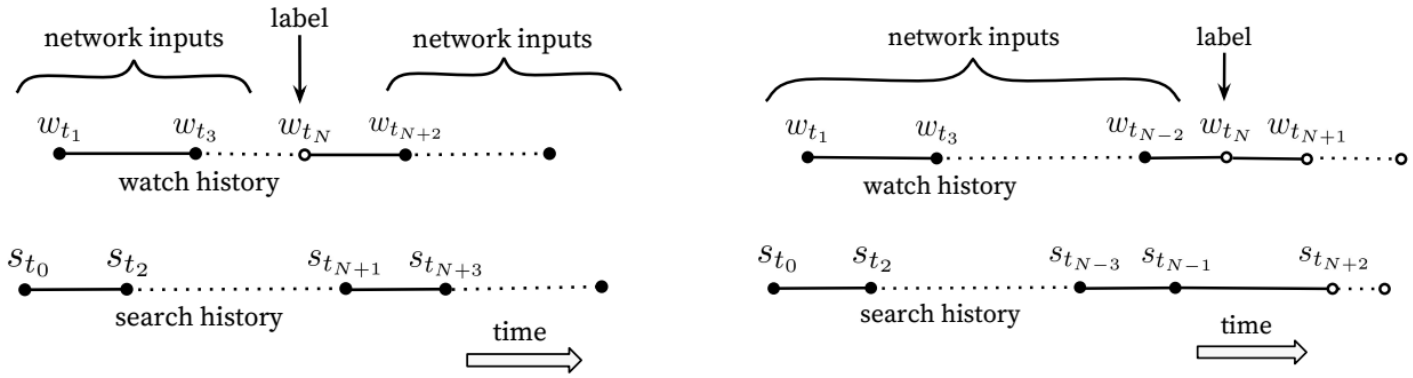
Figure 4: For a given video [26], the model trained with example age as a feature is able to accurately represent the upload time and time-dependant popularity observed in the data. Without the feature, the model would predict approximately the average likelihood over the training window.



3.4 Label and Context Selection

- Training examples are generated from all YouTube watches rather than just watches on the recommendations we produce. (Diversity)
- Generate a fixed number of training examples per user, effectively weighting our users equally in the loss function.

- Predict the user's next watch, rather than predict a randomly held-out watch (figure below).
- Discard sequence information and represent search queries with an **unordered** bag of tokens.

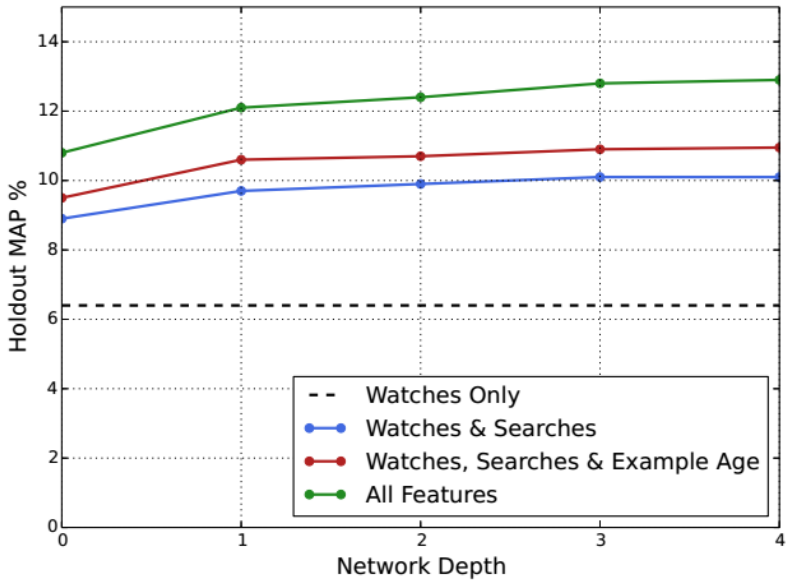


(a) Predicting held-out watch

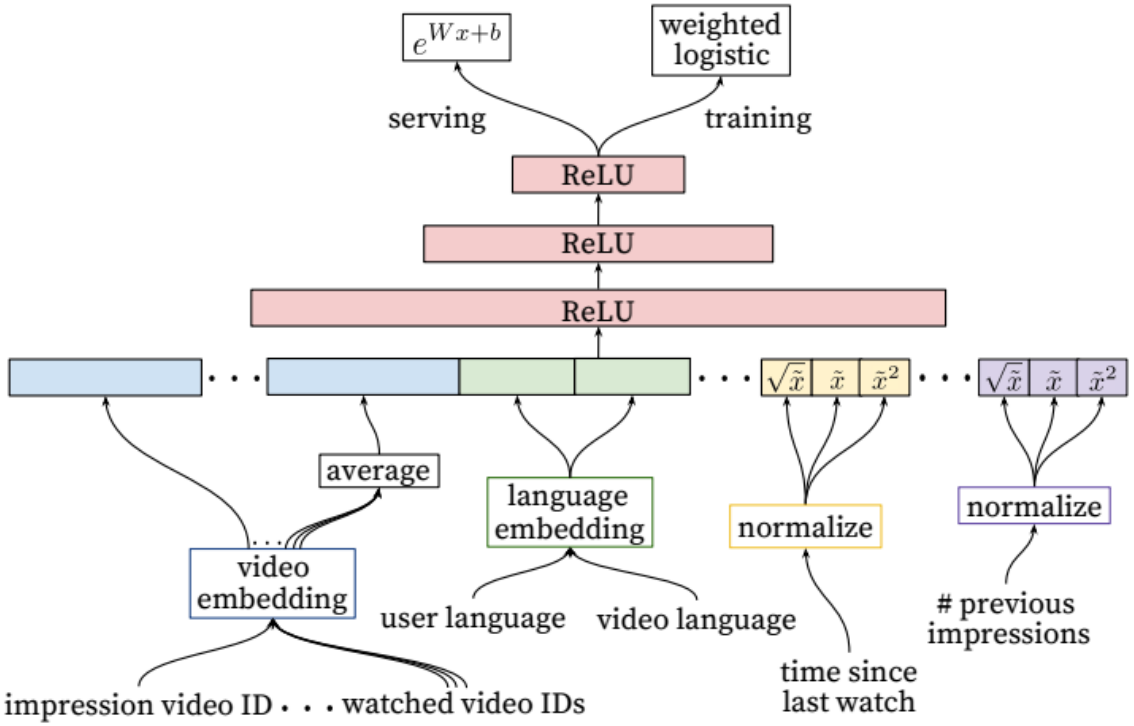
(b) Predicting future watch

3.5 Experiments with Features and Depth

- Depth 0: A linear layer simply transforms the concatenation layer to match the softmax dimension of 256
- Depth 1: 256 ReLU
- Depth 2: 512 ReLU $\rightarrow$ 256 ReLU
- Depth 3: 1024 ReLU $\rightarrow$ 512 ReLU $\rightarrow$ 256 ReLU
- Depth 4: 2048 ReLU $\rightarrow$ 1024 ReLU $\rightarrow$ 512 ReLU $\rightarrow$ 256 ReLU



4 Ranking

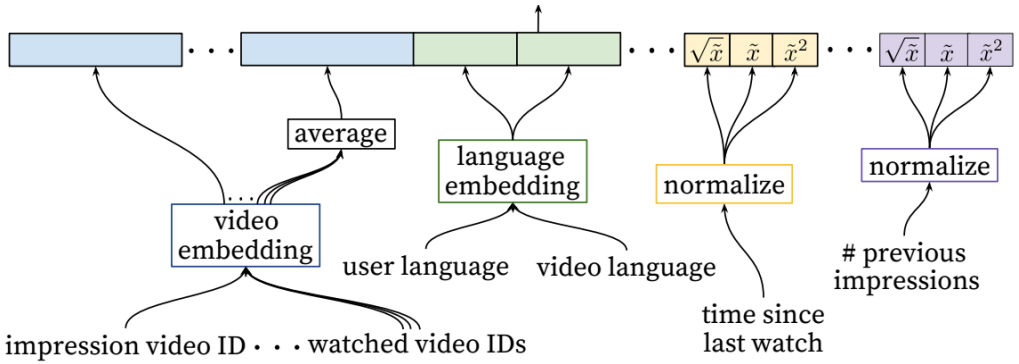


Ranking by click-through rate often promotes deceptive videos that the user does not complete (“clickbait”) whereas watch time better captures engagement.

4.1 Feature Representation

Embedding Categorical Features

- Use embeddings to map sparse categorical features to dense representations suitable for neural networks.
- Very large cardinality ID spaces (e.g. video IDs or search query terms) are truncated by including only the top N after sorting based on their frequency in clicked impressions. Out-of-vocabulary values are simply mapped to the zero embedding.
- As in candidate generation, multivalent categorical feature embeddings are averaged before being fed-in to the network.



4.1 Feature Representation

Normalizing Continuous Features

- Neural networks are notoriously sensitive to the scaling and distribution of their inputs
- A continuous feature x with distribution f is transformed to $\tilde{x}$ by scaling the values such that the feature is equally distributed in $[0, 1)$
- In addition to the raw normalized feature quantiles $\tilde{x}$, we also input powers $\tilde{x}^2$ and $\sqrt{\tilde{x}}$, giving the network more expressive power by allowing it to easily form super- and sub-linear functions of the feature.

4.2 Modeling Expected Watch Time

- Goal is to predict expected watch time given training examples
- The model is trained with logistic regression under cross-entropy loss. The positive (clicked) impressions are weighted by the observed watch time on the video. Negative (unclicked) impressions all receive unit weight.
- The odds learned by the logistic regression are $\frac{\sum T_i}{N-k}$.

$$\frac{\sum T_i}{N-k} = \frac{\frac{\sum T_i}{N}}{\frac{N-k}{N}} = \frac{E[T]}{1-P} = \frac{E[T](1+P)}{1-P^2}$$

Since P is small, this product is close to $E[T]$.

- For inference we use the exponential function e^x as the final activation function to produce these odds that closely estimate expected watch time.

Logistic Regression

$$y = \frac{1}{e^{-(w^T x+b)}}$$

$$\frac{1}{y} - 1 = e^{-(w^T x+b)}$$

Odds: $$\frac{y}{1-y} = e^{w^T x+b}$$

4.3 Experiments with Hidden Layers

- Experiments of the different hidden layer configurations
- If the negative impression receives a higher score than the positive impression, then we consider the positive impression's watch time to be *mispredicted watch time*.
- Weighted, per-user loss is then the total amount *mispredicted watch time* as a fraction of total watch time over held-out impression pairs.

Hidden layers	weighted, per-user loss
None	41.6%
256 ReLU	36.9%
512 ReLU	36.7%
1024 ReLU	35.8%
512 ReLU → 256 ReLU	35.2%
1024 ReLU → 512 ReLU	34.7%
1024 ReLU → 512 ReLU → 256 ReLU	34.6%

5 Conclusion

- YouTube deep neural network architecture for recommending videos, split into two distinct problems: **candidate generation** and **ranking**.

- The deep collaborative filtering model outperforming previous matrix factorization approaches used at YouTube
- **Classifying a future watch** to perform well on live metrics by capturing asymmetric co-watch behavior and preventing leakage of future information.
- **Withholding discriminative signals** from the classier was also essential to achieving good results.
- **Using the age of the training example** as an input feature removes an inherent bias towards the past and allows the model to represent the time-dependent behavior of popular of videos.

5 Conclusion

- Ranking is a more classical machine learning problem yet our deep learning approach outperformed previous.
- Deep neural networks require special representations of categorical and continuous features which we transform with **embeddings** and **quantile normalization**, respectively.
- Logistic regression was modified by **weighting training examples**, allowing us to **learn odds that closely model expected watch time**. This approach performed much better on watch-time weighted ranking evaluation metrics compared to predicting click-through rate directly.